

# 嵌入式 Linux 导航电子地图数据的组织及应用

周 巍 沈永增 王 燕 姚萌萌 (浙江工业大学 信息工程学院 浙江 杭州 310023)

**摘 要:** 为满足嵌入式系统中电子地图显示、导航的需要,研究了电子地图的数据组织与存储,提出采用地理数据分块实现数据的存储,并针对地理数据的存储方法所造成的数据不连贯性,研究了数据断点的处理方法。试验结果表明,研究的数据动态组织方法大大提高了地图显示的速度及数据连贯性。

**关键字:** 嵌入式;数据组织;数据断点;地图显示;数据连贯性

## Data Organization of Digital Navigating Map on Embedded Linux OS

ZHOU Wei, SHEN Yong-Zeng, WANG Yan, YAO Meng-Meng

(College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China)

**Abstract:** Data organization and storage of digital map are studied for the demand of digital map display and navigating service in embedded systems. A map data method is put forward to realize the data storage. Due to data inconsistency in map data storage method, the approach of handling data breakpoints is studied. Experimental results show that dynamic data organization method greatly improves the speed and data consistency of in map display.

**Keywords:** embedded; data organization; data breakpoints; map display; data consistency

电子地图显示作为其他功能的基础,是智能车辆导航定位系统中最重要的功能之一。系统中,地图刷新频率高,实时性要求强,数据容量大,然而常用的地图数据组织方法和地图显示策略通常是针对服务器或 PC 机的,对于小内存、低硬件性能的嵌入式系统很难适用。为此,不同的学者分别研究了地图数据的分块存储<sup>[1]</sup>和分区索引<sup>[2,3]</sup>等方法,以加快地图显示的速度。本文通过多文件存储数据的方法来加快地图显示速度,并针对使用多文件后所造成的数据不连贯性问题进行了研究解决,增加电子地图的直观性。

### 1 导航电子地图物理存储方法

移动终端上的导航电子地图数据的物理存储与组织,可以采用 2 种方法实现 嵌入式数据库和文件系统。

#### 1.1 基于嵌入式数据库的方法

嵌入式数据库系统是指支持移动计算或某种特定

计算模式的数据库管理系统,经过不断的发展,嵌入式数据库领域已经开始进入实用化和产品化的阶段,然而嵌入式数据库仍在发展和完善的过程中,其产品的功能、规模远不及传统数据库成熟。而且目前嵌入式数据库的研究还集中在完善数据库本身,如:数据复制、移动事务处理、数据的安全性、实时性要求、数据广播和 XML 嵌入式数据库等。嵌入式空间数据库的研究缺少像 PC 机上 ArcSDE 这样成熟、实用的空间数据引擎。目前的嵌入式数据库管理系统并不能完全满足其需求,但未来随着其自身技术的发展和完善,可能会成为导航应用中数据存储的主要方式之一。

#### 1.2 基于文件系统的方法

目前,国内外导航系统使用的导航数据在移动客户端主要还是采用文件存储,如国际流行的 Kiwi<sup>[4]</sup>, SDAL, GDF 等,以及各家导航公司自定义的数据格式。

文件系统用于导航地图数据存储时具有明显的优

点，如：支持大数据量的文件；支持各类数据文件的存储，数据格式不受限制。但也有其缺点，如数据访问速度比较慢，尤其对大数据量的访问。所以在结合文件系统存储数据优点的基础上，本文进一步研究了导航数据的物理存储采用多文件存储，来进行内外存之间的数据交换。这种多文件存储方法，不仅提高了地图的显示速度，而且数据访问速度快，通过文件内部偏移量进行数据的读取可以达到很高的读取效率。

## 2 物理存储的逻辑结构

数据组织的文件框架如图 1 所示，导航数据根据功能不同分为层间和层内的数据组织关系，同时将不同尺度的导航数据采用不同的地理层表示，不同尺度的地理层形成了导航数据纵向方向上的分层。在某一个横向方向上即针对某一地理层，按照一定的规则，将地理层划分为小块数据组织。

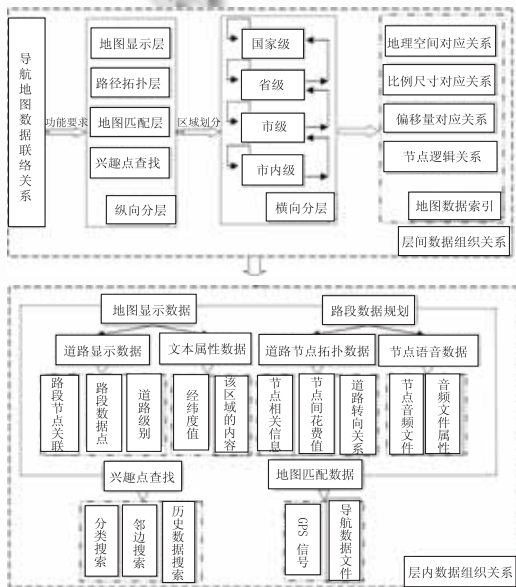


图 1 导航地图数据文件组织框架

## 3 空间数据显示数学模型分析

假设某矢量存储的空间数据占用物理存储量为  $M$ ，数据读入内存速度为  $V_1$ ，显示速度为  $V_2$ ，数据读取准备时间为  $f(n)$ ，用户可以等待的时间为  $T$ ，S3C2440 开发板的内存为  $D$ ，则有以下公式成立：

$$M(1/V_1 + 1/V_2) + f(n) \leq T \quad (M < D)$$

在开发板内存大小一定的情况下，数据读取准备时间一定时，若  $V_1, V_2, T$  为常数，必须减少  $M$  才

能满足数据显示和处理要求。由于任意时刻屏幕显示的图形数据只是读入数据  $M$  的一部分，因此，适当减少  $M$  的非屏幕显示区域的数据，并不影响屏幕图形数据的显示。本文将空间矢量数据  $M$  分为  $N$  份，任意时刻在开发板上显示图形数据的时候，只是读取部分图形数据，来满足快速显示图形的要求和数据存储需要。

## 4 地理数据分块原理

把一个大文件中的数据根据用途将其分割成多个文件，则首先需要对这个大文件中的地理数据进行分块，然后根据需要再将分块数据重组成适当的小文件分别保存。由于每个区块的坐标是确定已知的，所以通过分层的区块组织在引用空间数据时可以根据相应的部件位置在不同的区块中查找，从而避免了在全部数据中进行遍历。可以把块的大小设为硬件屏幕所能显示的大小，这样一屏刚好可以显示一块的数据内容，使读入数据最经济。当然也可以只读取一个主区块和相邻八个区块的数据<sup>[5]</sup>，通过实时拷屏的方式提高显示速度，这样做会比通过遍历记录集和空间索引的方式快许多。由图中确定地图区块号后，直接可以通过文件指针偏移量找到相应的文件区块进行读取。

### 4.1 矢量图形地理数据的分块

嘉兴地图用 Mapinfo 工具提取出数据之后，由于整个嘉兴的道路数据都存在了，所以数据量庞大，如果对数据不加处理，直接用于显示，那么在平移电子地图时，界面显示不流畅。

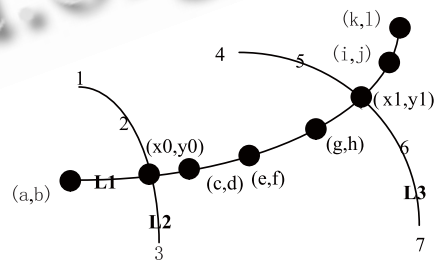


图 2 相交路段示意图

下面就是对地理数据进行处理的过程。如图 2 所示，图由 3 条路段  $L_1, L_2$  与  $L_3$  相交组成，其中点  $(a,b)$  与点  $(k,l)$  是路段  $L_1$  上的边界点，点  $(c,d)$ 、点  $(e,f)$ 、点  $(g,h)$  与点  $(i,j)$  是路段  $L_1$  上的中间点，点  $(x_0, y_0)$  与点  $(x_1, y_1)$  分别是路段  $L_1$  与  $L_2, L_3$  相交的点。

相交路段上的数据进行分块实现的过程如下：

把 L1 作为待匹配的路段，L2、L3 作为与 L1 匹配的后选路段。

先将 L1 上从点(a,b)到点(c,d)的路段与 L2 上的点匹配。如果两组点不在同一条路段上且连成的线段相交，则进行分段，执行  $\text{count}++$ ，否则执行  $\text{count}++$ 。

L2 上的 1 - 2 路段，由于与(a,b)、(c,d)路段没有交点，所以在 L2 的路段上查找下一路段 2 - 3。如果两个条件还未符合，继续取 L2 上的下一组点，继续执行  $\text{count}++$  过程，直到取完路网上的数据。

分段后的点(a,b)、点(x0,y0)保存在文件中，然后把点(x0,y0)作为下一数据分段的起点，继续重复  $\text{count}++$  过程，直到取完路网上的数据。

这样，整个大数据块经过处理后，就可以容易地分割路段上的相交节点，但是这样造成了数据衔接问题上的不连续性。

#### 4.2 地理数据断点的处理

对路段上的数据进行分块后，在个别处会出现断点现象，使得分块后的数据不能保持连贯性，造成完整的一条道路出现断路现象。针对这样的问题，就需要对数据分块后的断点进行处理。图 3 是处理拓扑函数交点的断线图，其中点(x0,y0)、(x1,y1)、(x2,y2)、(x3,y3)、(x4,y4)、(x5,y5)是每条路段上的边界点。

假设刚才分块后保存的数据格式如下：

```
num1
x0 y0
.....
x1 y1
num2
.....
```

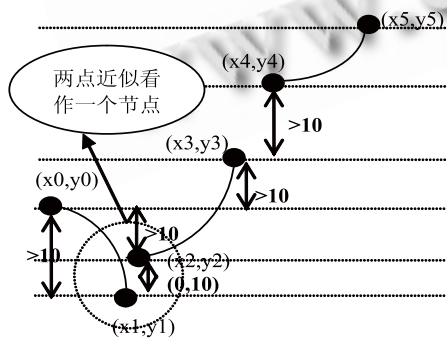


图 3 路段边界示意图

首先定义两个数组 m[5000]、n[5000]，用作存放每条路段的边界点。设置全局变量 int count，用作

计数。fromID、toID 也是起到计数作用，分别记录起始点编号与终点编号。执行步骤如下：

读取第一组(num1)数据，开始赋值  $m[\text{count}]=x0;n[\text{count}]=y0;m[\text{count}+1]=x1;n[\text{count}+1]=y1$ ;(当  $\text{count}=1$  时，即  $m[1]=x0;n[1]=y0;m[2]=x1;n[2]=y1$ )。首先记录  $\text{fromID}=\text{count} = 1;\text{count}++$  变成  $\text{count}=2$ 。

处理 num1 数据中的下一个边界点(x1,y1)，与前面的节点进行几何距离计算，由图 3 可知，该两点间的距离不在 (0,10) 范围之间，所以  $\text{toID}=\text{count}=2,\text{count}++$  变成  $\text{count}=3$ 。这里选择的 (0,10) 区间可以是其他任意的小范围，只是作为判断两节点是否可以近似看作一个节点的长度依据。

读取第二组(num2)数据，经过上一轮数据处理之后，此时  $\text{count}=3$ ；赋值  $m[\text{count}]=x2;n[\text{count}]=y2;m[\text{count}+1]=x3;n[\text{count}+1]=y3$  (即  $m[3]=x2;n[3]=y2;m[4]=x3;n[4]=y3$ )，记录  $\text{fromID}=\text{count}=3;\text{count}++$  变成  $\text{count}=4$ ；计算点(x2,y2)与前面节点间的距离，然后与某个设定值，例如 10 进行比较，如果小于 10，则 fromID 减 1，count 也减 1，相当于该节点与上一节点近似看作同一个节点，图 3 中用虚线的圆圈出来的即是，这样巧妙地处理了数据断点问题。

把 num2 数据中的下一个边界点(x3,y3)往前移，记录  $m[\text{count}]=x3;n[\text{count}]=y3$  (即  $m[3]=x3;n[3]=y3$ )， $\text{toID}=\text{count}=3$ ， $\text{count}++$  变成  $\text{count}=4$ ，点(x3,y3)再与前面的节点计算距离，然后与值 10 进行比较。

用同样的方法处理其他路段上的数据，直到结束。

下面两组数据，左边一列是数据分块后未经过数据处理的情况，右边一列是经过数据连贯性处理后的情况，如下所示：

1 2 num1 pen	1 2 num1 pen	
x0 y0	x0 y0	
.....	.....	
x1 y1	x1 y1	← (x1,y1)与(x2,y2)近似看作一个节点
3 4 num2 pen	2 3 num2 pen	
x2 y2	x2 y2	
.....	.....	
x3 y3	x3 y3	
5 6 num3 pen	4 5 num3 pen	
x4 y4	x4 y4	
.....	.....	
x5 y5	x5 y5	

图 4、图 5 是两幅比较图，图 4 显示的是数据分块后，断裂的数据点未经过处理；图 5 是经过数据点连贯性处理后的示意图。



图 4 数据点不连续图



图 5 数据点连续图

### 4.3 数据多文件存储

一个大文件经过上面两个步骤处理后，既解决了各块数据边界难分割的问题，又处理了数据边界分割后造成的数据不连贯性问题。但是大数据量在嵌入式设备上不可能一次性全部读入内存，必须根据显示屏的大小分文件。本文设计的文件大小要比开发板的屏幕稍大些，可以从三方面考虑：每次读取数据的量；一次性读取文件的个数；分块后的文件个数。如果选择的文件大小比屏幕小或者相等的话，那么显示电子地图时要涉及到多个文件。多个文件之间读取数据要比一个文件中读数据效率低，而且同样的算法要执行多次。

### 5 试验

本文用嘉兴市的地图数据进行了试验，系统硬件配置为：Samsung 的 S3C2440，该芯片采用的是

ARM920T 内核，主频 400MHZ，内存 64MB，分辨率为 480 × 272 像素。采用 SD 卡移动存储盘存储地图数据，在嵌入式平台上试验电子地图显示速度，并给出试验的性能参数，见表 1。

表 1 性能参数表

系统功能	参数	备注
地图显示	比例尺 6	根据地图显示的区域可设定不同的比例尺(1~39)
地图平移	分块前	12s 地图平移很慢，甚至有卡住现象，很久才有反应
	分块后	30ms 地图平移速度快，反应灵活

### 6 结语

导航地图数据的存储是导航系统的关键，本文分析了导航电子地图物理存储的 2 种实现方法：基于嵌入式数据库和文件系统。通过对比二者的优缺点，本文研究了地理数据分块存储数据的方法，且在这一方法的基础上避免了数据不连续的问题，从而使得电子地图在显示上速度明显加快又使画面保持连贯性。

### 参考文献

- 1 李鲁群,李成名,林宗坚.服务于 PDA 的矢量数据分块存储数据结构的研究.测绘学报, 2002,31(2):170 - 174 .
- 2 王轩,李鹤元.基于分区索引方法的地图快速显示的设计与实现.测绘通报, 2002,(10):56 - 61.
- 3 刘新贵,孙群,黄雅娟,李文.实现电子地图快速显示的策略和方法.测绘通报, 2004,(1):54 - 55.
- 4 刘有源,黄新明,陈定方,刘金鹏.KIWI 数据格式在导航系统中的应用研究.计算机辅助设计与图形学学报, 2003,15(4):503 - 508.
- 5 陈刚,贾奋励.超大数据量矢量电子地图显示的方法研究及实践.测绘通报, 2000,(2):15 - 17.